Configuration Management

in

Security related

Software Engineering Processes

Klaus Keus, Thomas Gast [*]
Bundesamt für Sicherheit in der Informationstechnik
Postfach 20 03 63, D - 53133 Bonn

[*] e-Mail: {gast, keus}@bsi.de

CONFIGURATION MANAGEMENT IN SECURITY RELATED SOFTWARE ENGINEERING PROCESSES

Klaus Keus, Th. Gast, Bundesamt für Sicherheit in der Informationstechnik (BSI), Postfach 20 03 63, D - 53133 Bonn, Germany

Abstract

IT-Security requires specific enhancements and tailoring during the complete life cycle of the product or system, including a security focused SW-engineering process. One of the key technologies to manage the software engineering process is the use of a tool driven Software Configuration Management.

Software Configuration Management (SCM)[10] is an aspect of establishing that the functional requirements and specifications are realised in the implementation during the whole life cycle. SCM is the activity of controlling the software product by managing the versions of all components and their relationships. It is one of the fundamental activities of software engineering in general and becomes most important in the development of high assurance software as in IT-Security.

This paper demonstrates, that the management of the whole software life cycle using SCM guarantees the traceability from the requirements specification (phase) via the design and the implementation phases to the final software product and maintenance phases by coordinating/controlling the changes in all phases of the software engineering process.

Using SCM with defined roles and access control enables the implementation of security measures to manage the software engineering process in a defined and controlled way. Thereby the assurance of the development process itself will be improved.

This paper discusses basic requirements of a Software Configuration Management System to suit the field of IT-Security. The scope of these requirements extends from quality standards, such as the ISO9000, to the specifics in general acccepted "IT-Security Evaluation Criteria", such as the ITSEC (Information Technology Security Evaluation Criteria)[6] and the CC (Common Criteria)[2]. A first approach to a maturity model for SCM in IT-Security will be given.

1. Introduction

Configuration Management delivers the key to transparency in the software engineering process. It is a precondition for successful manufacturing of high quality software products. Configuration Management becomes more and more important with increasing complexity of the software products and the software engineering process itself. Without a defined change and integration management there is no way to control of the software engineering process. Software Configuration Management (SCM) increases the quality and the assurance of the software engineering process with direct impact on the quality and assurance of the IT-products. How to run a SCM in the special environment of the development of critical software systems in general, and of high security software systems in special, will be discussed in the following chapters. Focus of the paper are the more technical aspects. Although the organisational and the management issues are very important and have to be respected in a more global and common view, these issues are not respected in this paper.

Chapter 2 will briefly introduce the general scope and functionality of Software Configuration Management. The terminology will be defined. It will be discussed what SCM really is, how it works and why we need it.

In chapter 3, general requirements for a SCM will be presented. The requirements will be derived from the quality standard ISO 9000 part 3 [8]. It deals with general models and criteria like the V-Model [13][VMOD] from the German KBSt, the Capability Maturity Model (CMM)[3] and the ESA Software Engineering Criteria [5]. These will be considered directly or indirectly. These requirements will be mapped to basic security requirements derived from the Information Technology Security Evaluation Criteria (ITSEC). Ways to fulfill these conditions and requirements will be discussed. A survey of the impact to the quality and assurance of the software engineering process and the software product will be given.

In chapter 4, special security requirements to a SCM will be presented. The requirements will be derived from the Information Technology Security Evaluation Criteria (ITSEC), and consideration will be given to the Common Criteria (CC), the Canadian Trusted Computer Product Evaluation Criteria (CTCPEC)[4] and the Security Engineering CMM. Possible ways to comply with these requirements will be offered. An analysis and survey of the impact on the assurance of the software engineering process and the software product will be presented.

In chapter 5 the state of the art in SCM will be explained. A SCM maturity level model will be introduced. The application of technical solutions using SCM tools will be discussed.

Chapter 6 will present some future aspects concerning the integration of SCM in the development process of high security software products. Some control approaches will be discussed to ensure the required impact of the actually installed SCM on the assurance of the software engineering process and the resulting software products, stretching from process and product evaluation to the accreditation of the entire software manufacturing process.

2. Software Configuration Management

The most frustrating software problems are well known by every software professional in the form of the reappearance of recently fixed bugs, mysterious disappearence of implemented and tested features, or even the total failure of a fully tested program. In our view such problems may lead to severe security problems if they are not discovered before shippment and installation. Therefore first we have to investigate where these problems derive from.

The increasing complexity of the software products shown in figure 2.1 has direct impact on the necessary coordination of the software manufactoring process. Many different people of possibly different institutions and/or locations may work on the same project. It is not possible and sufficient to build a rigid integration plan because the coordination of the parallel development activities must cover the complete software life cycle.

The coordination of designing such software products has to solve conflicts resulting from problems such as simultaneous update, shared code, common code, or multiple versions. The key is the qualified implementation and maintenance of a control system.

SCM reflects the current configuration and status of the software at any time, controls any changes to any object of the software, and maintains the traceability throughout the whole software life cycle. Also special security aspects, e.g. integrity, confidentiality and availability, are issues in the scope of SCM. SCM ensures the integrity and consistency of the software throughout the manufacturing process. SCM delegates the responsibility to check out the software modules to the developers, testers and so on in a defined way, to guarantee special confidentiality requirements on the "Need-to-Know"-principle in the manufacturing process of high security software components. SCM ensures the availability of any software configuration at any time.

SCM is the process of controlling the evolution of the software product by managing the versions of its components and their relationships. The key tasks of SCM can be summarized in configuration control, change management, and the management of a diversity of revisions, versions, deltas and conditional code.

SCM is a complex task that itself has to been driven in a well defined manner as shown by figure 2.1.

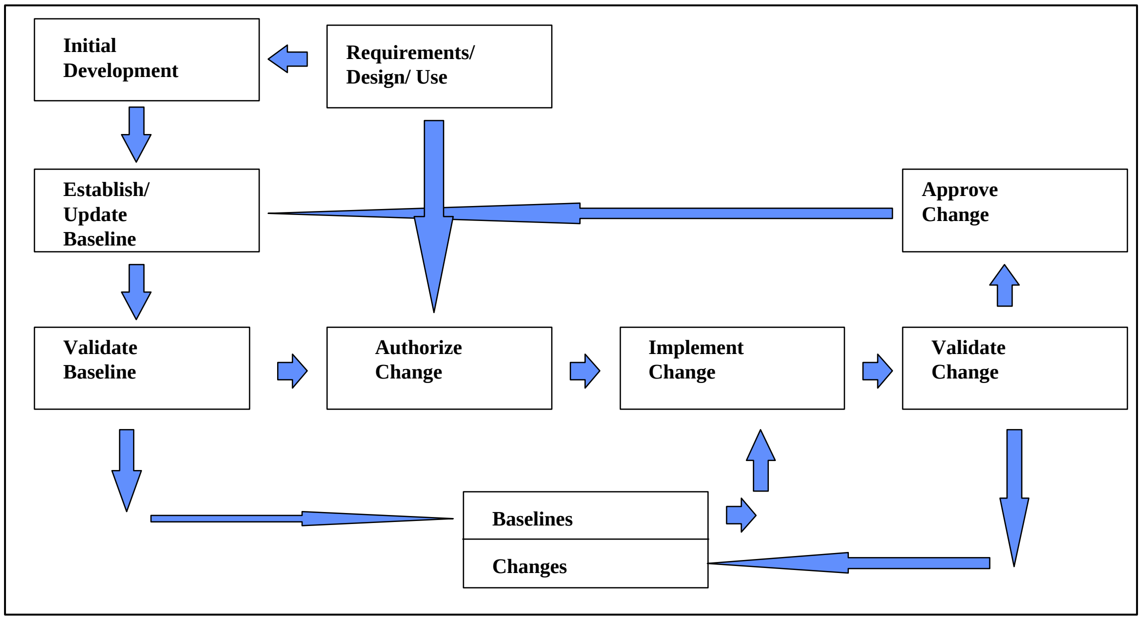

Figure 2.1 Well Defined SCM Process

To maintain the traceabiliy during all phases of the development process all outcoming documents from the requirements, the design to the implementation and testing must be managed by SCM. This is the precondition of keeping the validity, the quality and the assurance of the software engineering process. Every change request activates a complex implementation change control procedure as shown in figure 2.2.

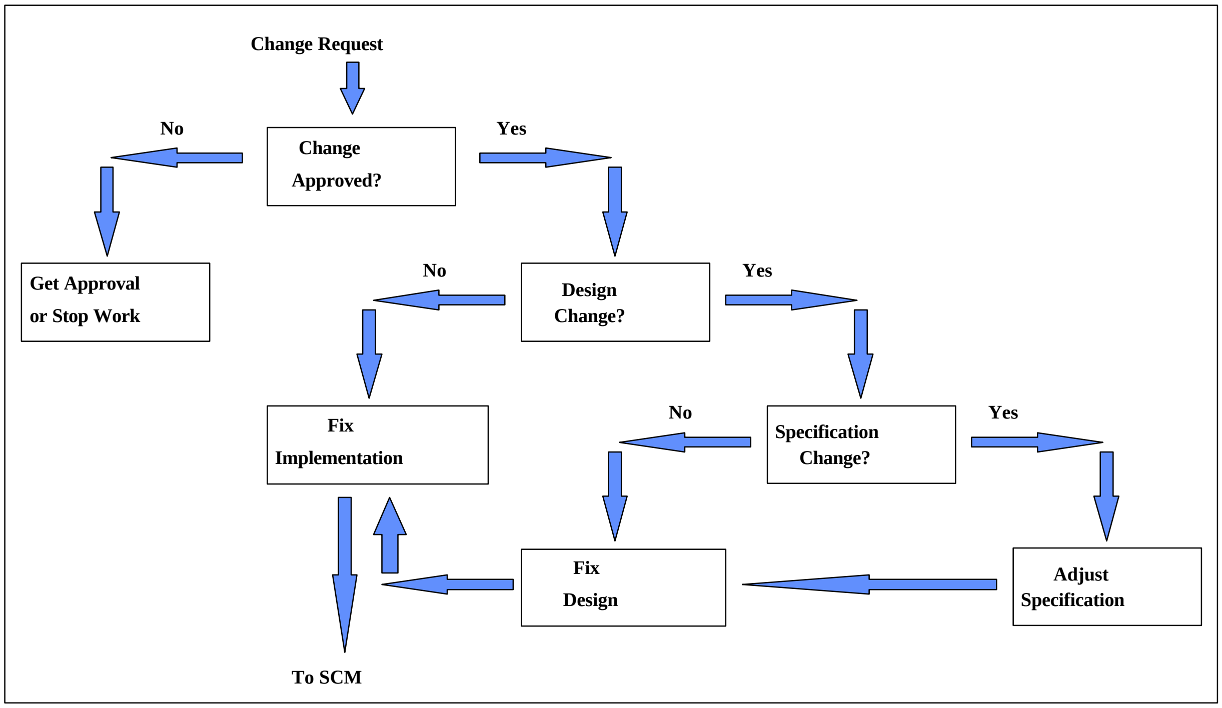

Figure 2.2 Well Defined Implementation Change Control Procedure

In the following chapters the requirements to the SCM will be refined to a more specific level with references to the quality and security assurance standards and criteria.

3. General Quality and Assurance Aspects of SCM

Quality aspects of the software manufactoring process in general are respected in the ISO 9000 Part 3 [8]. One of the main parts of this standard is chapter 6 dealing with configuration management. In the development of high security products it is important to analyze the impact of these requirements to the assurance of the manufactoring process and to the assurance of the software products themselves. Therefore the requirements of the ISO 9000 are taken in relation to the ITSEC if possible. The technical and administrative requirements concerning SCM are listed and the implementation of a tool based SCM to fulfill these requirements is discussed.

The first requirement to be discussed is to **identify uniquely the versions of each software item**. There is a link to the ITSEC requirement **The TOE (Target of Evaluation), its basic components and all documents provided ... shall possess a unique identifier**. It is a basic feature of SCM to guarantee an unambigious access to any managed Configuration Item (CI) at any time. This is a precondition to control the development process as well as the process of any high security software product. These two requirements are completed by **identify the versions of each software item which together constitute a specific version of a complete product** [ISO] and **the configuration list provided shall enumerate all basic components out of which the TOE is built** [ITSEC]. So SCM gives evidence at any time that in fact just the specified high security product is delivered and installed. These requirements for instance may be complied by the use of unique triple identifiers in the form **name:version:type**. SCM enforces the uniqueness of the identifier all over the complete project directory tree. To avoid any confusion it should be prevented to use different PATH directives for redundant identifiers in the project at any time. Hence any change of a CI leads to a new identifier of the unique version in the history of the CI.

SCM delivers transparency of the current state of the development process. It must **identify the build status of software products in development or delivered and installed** [ISO]. Similar to the requirements of ISO 9000 e.g. the ESA proclaims the three hierarchical state levels **Development**, **Controlled** and **Static** which reflect the main stations of the software life cycle. In the development process of security critical software products the granularity of the development status must be increased to the various roles in the development process in a balanced way. A mapping of the responsibilities to the current status should be possible at any time. The development status consists of 4 detail mappings: the working status maps to the activity of code development, the integrate status maps to integration testing, the SQA (Software Quality Assurance) status to the system level testing, and the released status as output status to the development process. This SCM, defined by role and status, fits best the IT security requirements for confidentiality, integrity, and availability, and leads to a high quality and high assurance software development process. SCM enforces each CI to respect this state machine. A Secure SCM (SSCM) manages the access to any CI in accordance to this development model.

An important aspect of SCM is the coordination of the various developing and changing activities of the components in the software product during the software engineering process. During the development process a complex network of dependabilities of the several modules and the objects will be built by the use of IMPORT/EXPORT features or via inheritance and granting. This network reflects the general access to data structures and functionality. To manage the consistency of this network during the whole life cycle SCM accepts a change not before its validity is confirmed and the effects to other CIs are identified and examined. This is an important requirement derived from the ISO as well as from the ITSEC to guarantee the integrity of the software product during any change process.

A most critical task of SCM is to **control simultaneous updating of a given software item by more than one person** [ISO]. Two kinds of simultaneous updating have to be distinguished: the updating of two independent versions of the CI resulting in two independent versions of the whole software product and the updating of the two independent change processes resulting in one version of the CI using controlled merging.

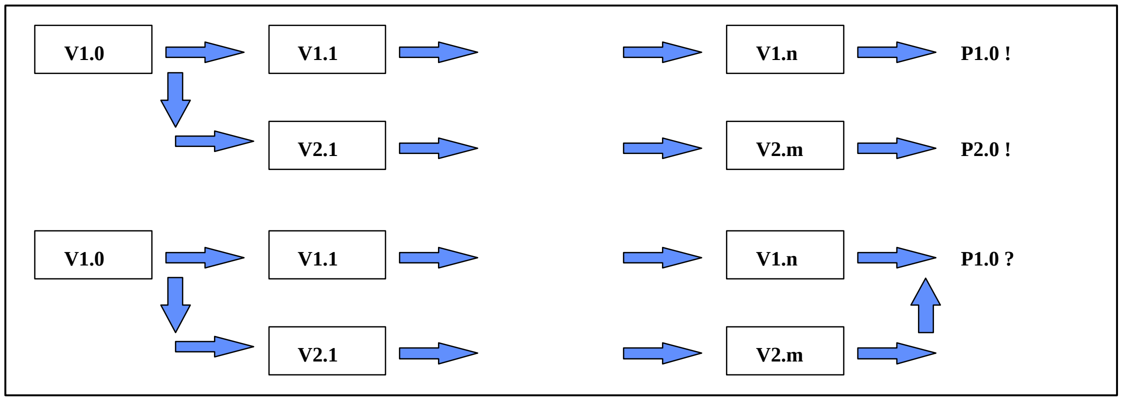

Figure 3.1 Branching; Branching and Merging

In the first case SCM has to guarantee the independence of the new branch throughout the whole life cycle of the two or more arising product versions. No merging is granted at any time. SCM provides a clear distinction between the version identifiers of the two branches like 1.n and 2.m shown in figure 3.1. SSCM is responsible for meeting the confidentiality aspects of those independent development activities.

The simultaneous updating of a CI - intending to merge two or more checked out versions to one version in the history tree of the CI - involves some special problems to SCM. How can SCM guarantee the correspondence of the simultaneous developer´s activities to avoid the occurrency of conflicts? How can SCM trace the history of the CI following a complex branch and merge tree when problems occur? How can SCM manage and control the different responsibilities of several developers who are working simultaneously on the same version of the same CI? Taking this approach all requirements for confidentiality, integrity and availability to a high assurance software engineering process will be compromised. A successive and complex branch and merge process is not manageable. This way of simultaneous updating must be rejected even at check out time and must be verified at check in. Otherwise the software engineering process will run out of control.

The control of any change process is the precondition for integrity and availability of any version of any CI. SCM controls the update process in the time frame from check out to check in to get the complete transparency of any change activity. SCM must **identify and track all actions and changes resulting from a change request, from initiation through to release**[ISO]. **The configuration control tools shall be able to control and audit changes between different versions of objects subject to configuration control**[ITSEC]. **All modifications of these objects shall be audited with originator, date and time**[ITSEC]. To fulfill these requirements a complete audit trail of any change must exist. This audit trail delivers SCM with the information of the event, the cause and the initiator of the change process. The implementation of any change request becomes identifiable and verifiable. The responsibilities of all steps in the change process are defined and documented. Following these requirements the history of any change is available to SCM.

As discussed in section 2 and shown by figure 2.1 the SCM process itself runs in a well defined manner. A configuration management plan (CMP) fixes all necessary administrative activities. The CMP defines **procedures to identify, document, review and authorize any changes to the software items ...**[ISO]. Following the ISO and the ITSEC requirements the CMP defines the responsibilities, the actions to be performed, the tools, techniques and methodologies, and it defines the baseline to check the CIs first under SCM. The CMP is part of the security policy of any high assurance development process.

SCM is applied in the whole life cycle of the software product. SCM must **maintain procedures for identifying software items during all phases, starting from specification through development, replication and delivery**[ISO]. **All objects created during the development process ... shall be subject to configuration control**[ITSEC].

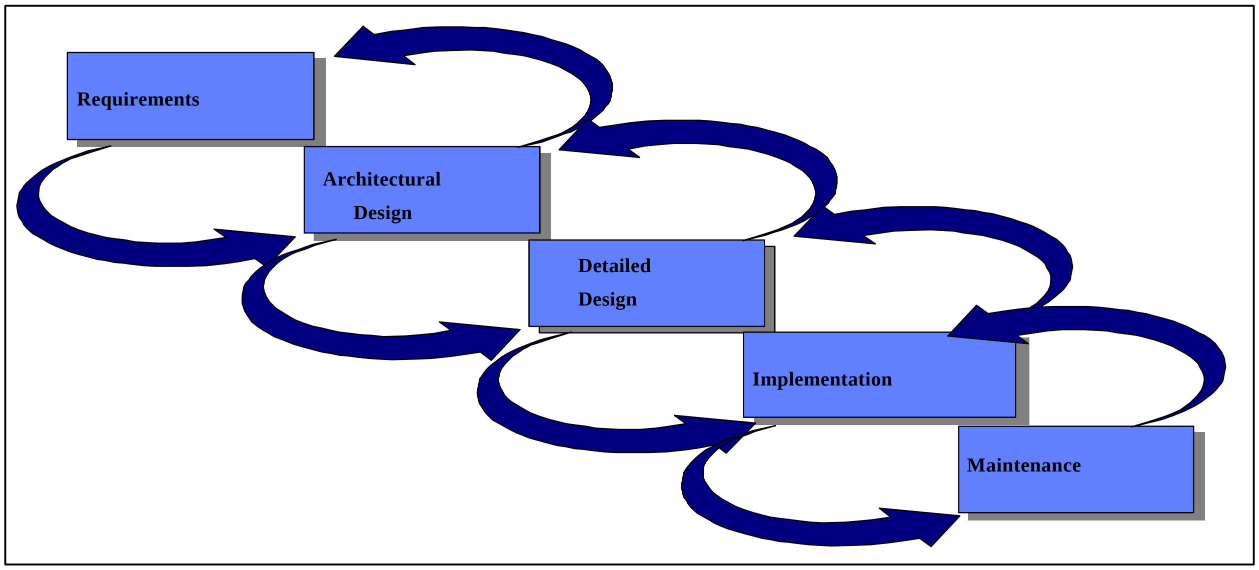

Figure 3.2 Traceability of the software engineering process

The traceability of the whole software life cycle is a precondition for the correctness of the software manufactoring process. SCM should manage all objects created during this life cycle. SCM identifies the correlations of the several versions of the specification documents following the object versions from the design, the implementation and the testing phase to the delivered version of the software product. SCM leads the development and maintenance process version driven through all phases of the software engineering process. The correlation of the several versions of the design documents of the product delivered to the according versions of the user manuals is dealed by SCM to guarantee the consistency of all product documents. This traceability aspect is the precondition of a successful security ITSEC evaluation of any software product. The scope of SCM includes more than the created objects during the engineering process. Even **all tools used in the development process shall be subject to configuration control**[ITSEC] to guarantee the rebuilding of any shipped version at any time. This includes the management of the applied CASE tools, compilers, debuggers, GUIs and operating systems. Also all external interfaces to the hardware or to software such as libraries must by under control of SCM. Following all these requirements SCM is able to guarantee the integrity and availability of any released version of the software product.

A SCM running in a defined way as shown in figure 2.1 fulfills the ISO quality requirements as discussed above and builds up a good base for the management of all objects in a high quality software engineering process. As shown in this chapter there is a close connection between the ISO quality and the ITSEC security requirements. Recognizing this aspect it becomes possible to install and to drive such a tool driven SCM which fulfills both aspects: the ISO quality and the basic ITSEC security requirements. So by the way we get a high quality SCM including basic features of a high assurance software engineering process and respecting the IT security aspects confidentiality, integrity and availability.

4. General Security Aspects of SCM

In this chapter specific security aspects of SCM are discussed. Security aspects that exceed the requirements from chapter 3 concerning the correctness of the software engineering process. SCM in the special scope of IT security deals with problems occuring by accidental or deliberate unauthorized modifications of software components.

Security in IT deals with the access of subjects (persons or processes) to objects (data, programs, IT in general) focusing to determined existing risks. These risks are called threats. To become independent from the technical development environment , e.g. from the operating system, special security functions are required and implemented in the SSCM itself. So SSCM provides a complete set of security functionality, at least it should include functions for identification and authentication, access control, accountability, audit and accuracy.

An independent identification and authentication is the precondition for administrating the different users of a role driven SSCM. SSCM administrates roles like developer, integration manager, SQA manager, build manager and system administrator. The developer is responsible for the development of the single modules. The CI is explicitly checked out to the private working area of the developer. The access to this CI by any other developer has to be prevented. The developer drives previous tests in his seperate development area (socalled α-tests, which primarly respect the functionality and the correctness aspects). The integration manager builds the final complete product by composing the several modules checked in by the developers in a step-by-step way. He is responsible for the integration testing and for the process of the construction of the current version of the product. The SQA manager is responsible for the final acceptance of the installed product version, including the responsibility for the final testing phase (socalled β-tests, including the quality tests for the product (functionality and correctness tests)). The current product version is checked to RELEASED. The build manager gets access to all released product versions and is responsible for the rebuilding of any released product version. Those CIs which have to be changed, will be checked out by the build manager. He initiates any change process by checking them out into the working area of the developers. The system adminstrator is responsible for the complete administration of the SSCM. This task includes the adminstration of the resources of the SSCM, the archiving of the CIs, the security aspects as e.g. user and access management and the maintenance of the SSCM system itself.

Based on this user administration the SSCM access control enforces a defined change process according to the state machine discussed in chapter 3. SSCM ensures **that only authorised changes by authorised persons are possible**[ITSEC]. In the development process of critical IT-products - e.g. as high security software systems - the access to security kernel modules must be restricted to a group of trustworthy developers. This approach supports a countermeasurement for possible confidentiality conflicts. This confidentiality aspect has to be guaranteed throughout all successive development activities respecting the increasing granularity concerning confidentiality requirements. Meeting these requirements it is necessary to have a detailled mapping of the development activities to the involved roles. This mapping of the different confidentiality levels can be managed e.g. following the Bell-LaPadula [1] approach. SSCM guarantees that **all security enforcing and security relevant objects ... shall be identified as such**[ITSEC]. SSCM rejects any unauthorized access to any CI. Subsequently each unauthorized access must be restricted or completely avoided and each of it has to be audited. The CI may be blocked until the SSCM administrator explicitely releases it. As this approach protects from malicious access it enables the SSCM to **ensure that the person responsible for acceptance of an object into configuration control was not one of its designers or**

developers[ITSEC]. SSCM guarantees the seperation of roles as a quality and security aspect in the development process.

The accountability is guaranteed by recording any action concerning the SSCM and especially any access to any CI. Any unauthorized access is marked seperately and is analyzed by an automatic audit and report process (e.g. "red light" or warning at the console).

SSCM provides an efficient audit mechanism to get extreme transparency of the complete life cycle of any CI. SSCM must **control and audit changes between different versions of objects ... . All modifications of these objects shall be audited with originator, date and time** [ITSEC]. Such an audit trail enables SSCM to manage the whole history of any CI. The responsibility of any change process is documented. At any time any malicious access or change to any CI shall be verified.

SSCM guarantees the accuracy of any CI and any released product version by providing a transparent and controlled change process. This approach is supported by the possibility of automatically rebuilding of any version of any CI or each released product. SSCM guarantees the accuracy and the integrity of the security kernel, regardless whether the kernel is concerned by the change process directly or indirectly. SSCM must **be able to identify all other objects ... affected by this change together with an indication of whether they are security enforcing or security relevant objects**[ITSEC]. Using this approach SSCM provides a special protection mode to guarantee the integrity of the security kernel of any software product.

The confidentiality, the integrity and the availability of all CIs and of the final released product versions will be ensured by SSCM following the discussed quality security aspects. Also additional topics as the limited reuse of highly confidential components may be managed by SSCM. Implementing the same security component only in a restricted number of software products will help to control the lack and its impact to all products using this component in the case of a successful penetration [12]. But this may work in contradiction to the economical aspects in SW engineering.

5. State of the Art

In current software engineering processes it may be distinguished between 5 maturity levels for software configuration management systems.

Level 1 is indicated by development processes without any configuration control. Such a process can not even guarantee the basic requirements of the ISO nor the ITSEC or the CC. Software products built by those processes should never be accepted in the scope of critical software in general and in the scope of high security software in special. A security evaluation of software products against the ITSEC or the CC would not be possible without any SCM.

Level 2 is indicated by a SCM without the application of a SCM tool. The SCM is managed completely by administrative activities. Such a SCM is characterized by high efforts to even guarantee the application versus the more basical quality and security requirements discussed in chapter 3. A non tool based SCM may be bypassed, ignored, or may be insufficient to prevent inconsistencies or unautorized modifications. Hence a security evaluation of a product using a SCM level 2 may be restricted to a lower assurance level of the ITSEC or CC (e.g. E1 respectively EAL1 or EAL2). But evidence has to given for each software product for the implementation and application of this administrativeable SCM. This kind of evidence may require high efforts in respect to every quality assurance and to every security evaluation process.

Level 3 is indicated by a tool based SCM, e.g. in the CC it is subleveled by partial or complete automation. In general these tools guarantee a defined SCM in the development process fulfilling the basic quality and security requirements discusssed in chapter 3. A tool based SCM is a precondition for a successful security evaluation of higher assurance levels (beyond E4 of the ITSEC respectively EAL 5 in the CC). If once verified the results of the security evaluation concerning SCM may be reused with minimum effort for all products managed by this tool based SCM. Most of these tool based SCMs however don`t fit the special security requirements discussed in chapter 4 and will not reach level 4 in our scale. Level 4 is indicated by such a tool based SSCM that completely fulfills the requirements explained in chapter 4. But even such level 4 SSCMs often dependent on the such security features of implemented environmental features as the underlying development operating system or the linked database.

Hence level 5 defines a SSCM of level 4 combined with full flexibility. Depending on the situation, the environment and the specific security related requirements the SSCM may be tailored including upgrading features in the sense of optimizing during the engineering process (e.g. including multilevel security (MLS) approaches). It has to be managed in a way independent of all the underlying or environmental SW-products as the operating system or the linked database. All the phases in the life cycle may be managed and controlled, the access control during the different phases may be adaptable from mandatory access control (MAC) to discrete access control (DAC). The reuse of highly confidential components has to be respected.

6. Future Aspects

In the development of high security software products as well as in the development of critical software in general, the evaluation and accreditation of the software development process including the infrastructure, the tools and the experience of the developers have become an established part of the software manufactoring process. Modern evaluation and accreditation approaches consider the impact of a sound development process on the security of an IT product/system. One basic part of this development process is SSCM. To minimize evaluation efforts future SSCMs should be independent of the specific operating systems and development environments. This approach enables a classification of the SSCM tools versus the requirements discussed in chapter 3 and 4 and will support the integration of the SSCM tools in modern developmental assurance approaches discussed in Europe and in USA. But as said in the introduction, the combination of technical aspects with the organisational and management issues will built a complete appropriate approach. This kind of a global approach is also valid for the SCM in IT-Security. So next steps are the interpretation and impacts from SCM to related issues in organisation and management.

The SSCM approach discussed in this paper currently finds high interest on the user and manufacturer side in the scope of high security software products, and leads to busy developmental activities of the tool manufacturers.

References

[1] [BLP] Secure Computer Systems: Unified Exposition and Multics Interpretation, D.E. Bell and L.J. LaPadula, The Mitre Corporation, 1976

[2] [CC] Common Criteria for Information Technology Security Evaluation - V1.0, CCEB, January 31, 1996

[3] [CMM] Key Practices of the Capability Maturity Model, Mark C. Paulk, Charles V. Weber, Suzanne M. Gracia, Mary Beth Chrissis, Marilyn Bush, Software Engineering Institute, Pittsburgh, 1993

[4] [CTCPEC] The Canadian Trusted Computer Product Evaluation Criteria V3.0e, Canadian System Security Centre - Communications Security Establishment - Government of Canada, 1993

[5] [ESA] ESA Software Engineering Standards, Issue 2, ESA Board for Software Standardisation and Control, Paris, 1991

[6] [ITSEC] Information Technology Security Evaluation Criteria, ECSC-EEC-EAEC, Brussels * Luxembourg, 1991

[7] [ITSEM] Information Technology Security Evaluation Manual, ECSC-EEC-EAEC, Brussels * Luxembourg, 1994

[8] [ISO] EN ISO 9000 Part 3: Guidelines for the application of ISO9001 to the development, supply and maintenance of software, NQZS, Beuth Verlag GmbH, Berlin, 1994

[9] [MSWP] Managing the Software Process, Watts S. Humphrey, Addison-Wesley Publishing Company, 1989

[10] [SCM] Software Configuration Management - Coordination for Team Productivity, Wayne A. Babich,

[11] [SEPA] Software Engineering - A Practitioner´s Approach, Roger S. Pressman, Mr Graw-Hill Book Company

[12] [TSRR] Trusted Software, Repositories and Reuse, Mark O. Aldrich, GRC International, Prodeedings of the 11. Annual Computer Security Application Conference, 1995 New Orleans, IEEE Computer Society Press, ISSN 1063, ISBN 0-8186-7159-9, page 208-216

[13] [VMOD] Planung und Durchführung von IT-Vorhaben in der Bundesverwaltung - Vorgehensmodell, KBSt, Bundesanzeiger Verlagsgesellschaft mbH, Köln, 1992